Business Analytics

1. Prediction, 1.2 Simple Models

Lars Schmidt-Thieme

Information Systems and Machine Learning Lab (ISMLL)
University of Hildesheim, Germany

Outline

Outline

Overall Procedure

given:

1. data set
2. target variable
3. loss

procedure:

1. **split the data** into a training and a test set.
2. **learn a model** from the training data.
3. **predict with the model** for the test data
   (withholding the target variable)
4. **evaluate the model** by comparing the true (withhold) values of the target variable and the predicted ones.

Constant Model

Generally, a model is a **function**

$$\hat{y} : \mathcal{X} \rightarrow \mathcal{Y}$$

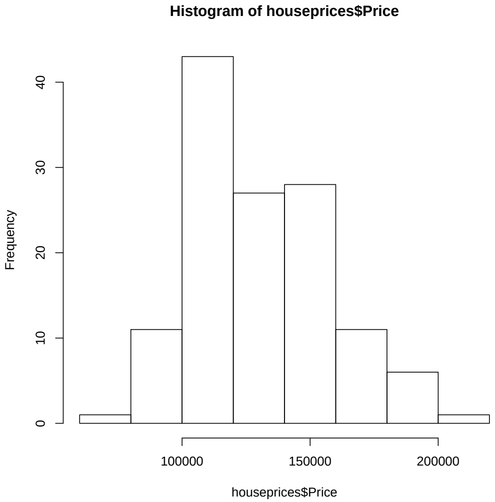
predicting different target values for different predictor values ("conditionally on predictors").

In section 1.1, our model has been a **constant**:

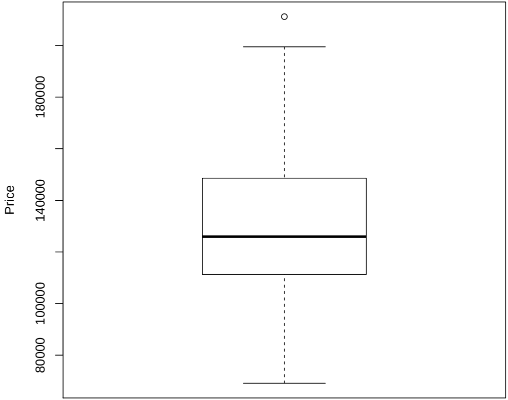
$$\hat{y}(x) := \hat{y} \in \mathcal{Y}$$

i.e., we predict the same value for **all** instances ("unconditionally")

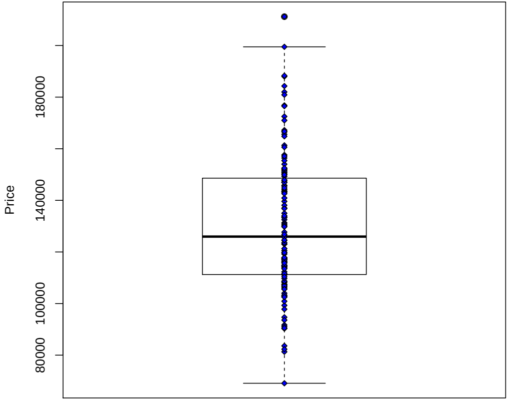
Example: House Prices (Histogram)



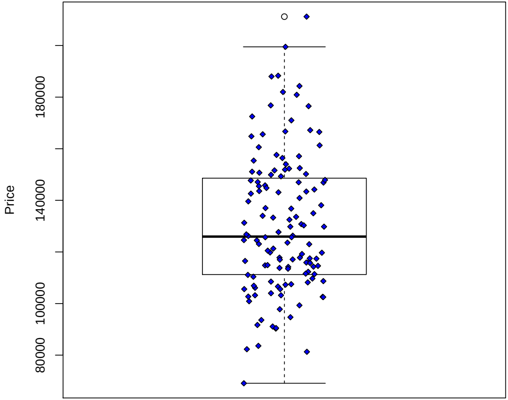
Example: House Prices (Boxplot)



Example: House Prices



Example: House Prices



Boxplots

A compact (one-dimensional) representation of a sample Y of a continuous variable:

Boxplots

A compact (one-dimensional) representation of a sample Y of a continuous variable:

- the **y axis** represents the **domain** of Y,

Boxplots

A compact (one-dimensional) representation of a sample Y of a continuous variable:

- the **y axis** represents the **domain** of Y,
- the **box** represents the
  - **first quartile** (bottom of the box)
    i.e., the value $y_1 \in \mathcal{Y}$ with
    $$|\{y \in Y \mid y < y_1\}| = \lfloor \frac{1}{4}|Y| \rfloor$$
  - **third quartile** (top of the box)
    i.e., the value $y_3 \in \mathcal{Y}$ with
    $$|\{y \in Y \mid y < y_3\}| = \lfloor \frac{3}{4}|Y| \rfloor$$

Boxplots

A compact (one-dimensional) representation of a sample Y of a continuous variable:

- the **y axis** represents the **domain** of Y,
- the **box** represents the
  - **first quartile** (bottom of the box)
    i.e., the value $y_1 \in \mathcal{Y}$ with
    $$|\{y \in Y \mid y < y_1\}| = \lfloor \frac{1}{4}|Y| \rfloor$$
  - **third quartile** (top of the box)
    i.e., the value $y_3 \in \mathcal{Y}$ with
    $$|\{y \in Y \mid y < y_3\}| = \lfloor \frac{3}{4}|Y| \rfloor$$
- the **line inside the box** represents the
  - **median**
    i.e., the value $y_2 \in \mathcal{Y}$ with
    $$|\{y \in Y \mid y < y_2\}| = \lfloor \frac{1}{2}|Y| \rfloor$$

Boxplots (2/2)

- ...
- the **whiskers** represent
  - **the smallest sample exceeding the lower fence** (bottom whisker) i.e., the value

$$y_0 := \min\{y \in Y \mid y > y_1 - 1.5(y_3 - y_1)\}$$

  - **the largest sample below the upper fence** (top whisker) i.e., the value

$$y_5 := \max\{y \in Y \mid y < y_3 + 1.5(y_3 - y_1)\}$$

Note: Upper fence $= y_3 + 1.5\text{IQR}$, lower fence $= y_1 - 1.5\text{IQR}$, IQR $=$ inter quartals range $= y_3 - y_1$.

Boxplots (2/2)

- ...
- the **whiskers** represent
  - **the smallest sample exceeding the lower fence** (bottom whisker) i.e., the value

$$y_0 := \min\{y \in Y \mid y > y_1 - 1.5(y_3 - y_1)\}$$

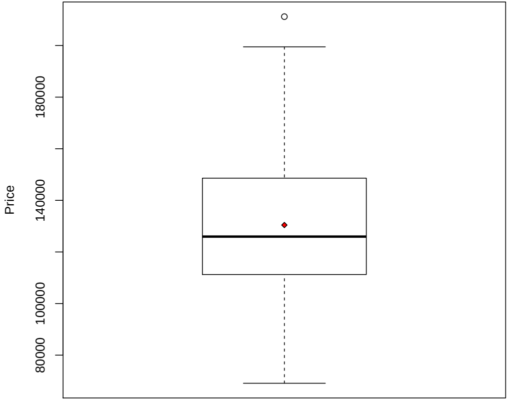
  - **the largest sample below the upper fence** (top whisker) i.e., the value

$$y_5 := \max\{y \in Y \mid y < y_3 + 1.5(y_3 - y_1)\}$$

- **points outside the whiskers** represent
  - **all samples below the lower fence** and
  - **all samples above the upper fence**.

Note: Upper fence $= y_3 + 1.5\text{IQR}$, lower fence $= y_1 - 1.5\text{IQR}$,
IQR $=$ inter quartals range $= y_3 - y_1$.

Example: House Prices (Boxplot w. Mean)



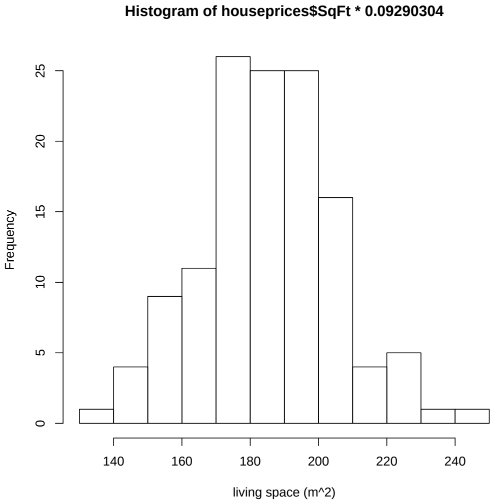
Note: Sometimes means are marked in boxplots, too (here: red diamond).

Example: House Prices: Predictors

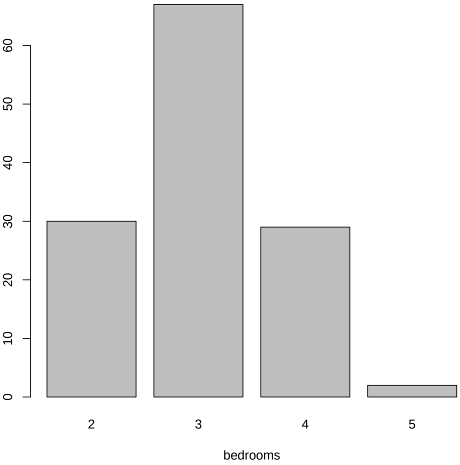
HomeID	Price	SqFt	Bedrooms	Bathrooms	Offers	Brick	Neighborh
1	114300	1790	2	2	2	No	East
2	114200	2030	4	2	3	No	East
3	114800	1740	3	2	1	No	East
4	94700	1980	3	2	3	No	East
5	119800	2130	3	3	3	No	East
6	114600	1780	3	2	2	No	North
7	151600	1830	3	3	3	Yes	West
8	150700	2160	4	2	2	No	West

Example: House Prices: Predictors

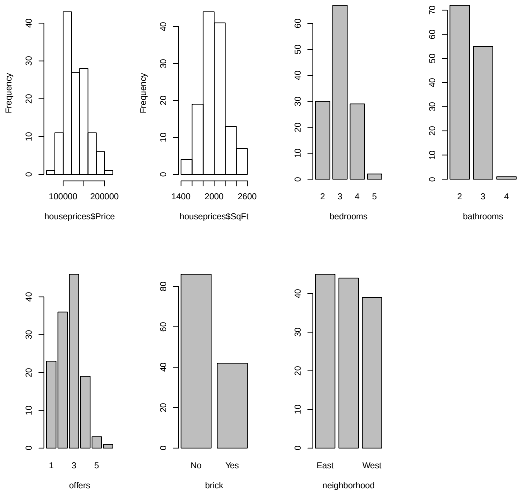
Example: House Prices: Predictors



Example: House Prices: Predictors



Example: House Prices: Predictors

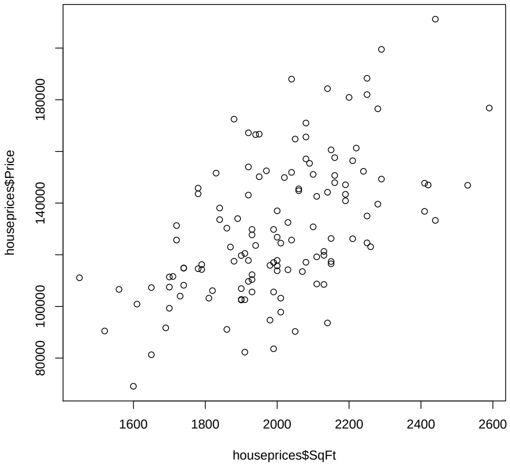


Scatterplots

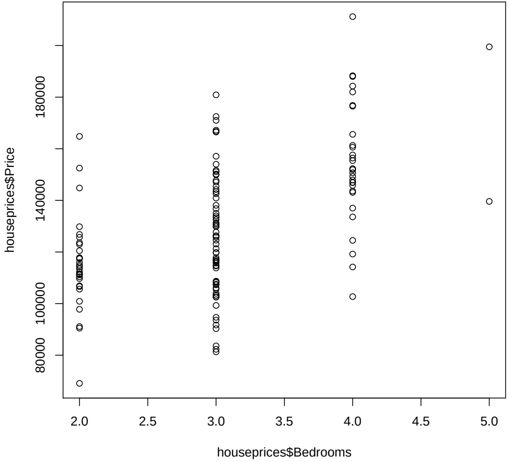
To visualize dependencies between a continuous target Y and a continuous predictor X within a sample $\mathcal{D}$, one can plot a scatterplot of Y vs X, i.e., points

$$\pi_{Y,X}(\mathcal{D}^{\text{train}})$$

Example: House Prices: Target vs Single Predictor



Example: House Prices: Target vs Single Predictor



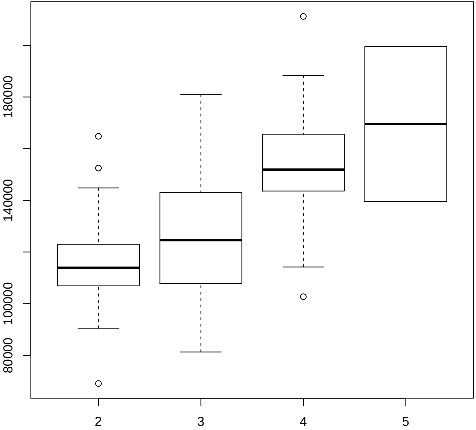
Conditional Boxplots / Grouped Boxplots

To visualize dependencies between a continuous target Y and a nominal predictor X within a sample $\mathcal{D}$, one can plot a boxplot per **group**, i.e., subset of the sample having the same value for the predictor:

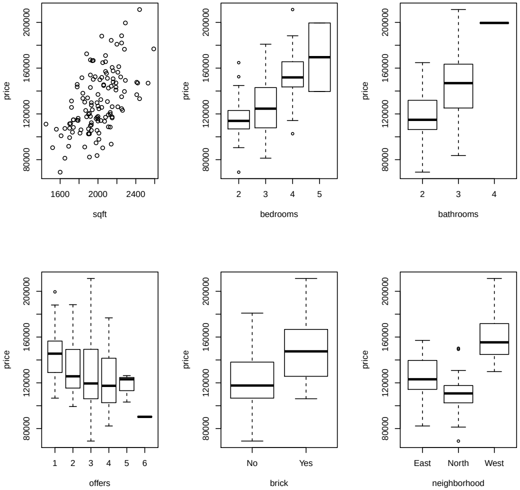
$$\pi_Y(\mathcal{D}^{\text{train}}|_{X=x}), \quad \text{for } x \in \operatorname{dom} X$$

$$\begin{aligned} \text{with } \mathcal{D}|_{X=x} :=& \{(x', y) \in \mathcal{D} \mid x' = x\}, \\ \pi_Y \mathcal{D} :=& \{y \in \operatorname{dom} Y \mid (x, y) \in \mathcal{D}\} \end{aligned}$$

Example: House Prices: Target vs Single Predictor (2/2)



Example: House Prices: Target vs Single Predictor (2/2)



Conditionally Constant Models

The most simple way to capture such a dependency between a target Y and a nominal predictor X is to build a **separate constant model for each group**, i.e., for each value of X.

- to optimize RMSE, one computes the group means:

$$\hat{y}(x) := \text{mean}\, \pi_Y(\mathcal{D}^{\text{train}}|_{X=x})$$

- to optimize MAE, one computes the group medians:

$$\hat{y}(x) := \text{median}\, \pi_Y(\mathcal{D}^{\text{train}}|_{X=x})$$

Example: House Prices: Conditionally Constant Models

X	RMSE($\mathcal{D}^{\text{train}}$)
—	28 035.64
Bedrooms	24 194.59
Bathrooms	—
Offers	27 401.65
Brick	24 350.58
Neighborhood	18 056.69

Note: 50:50 train/test split. By Bathrooms fails due to empty cell in train.

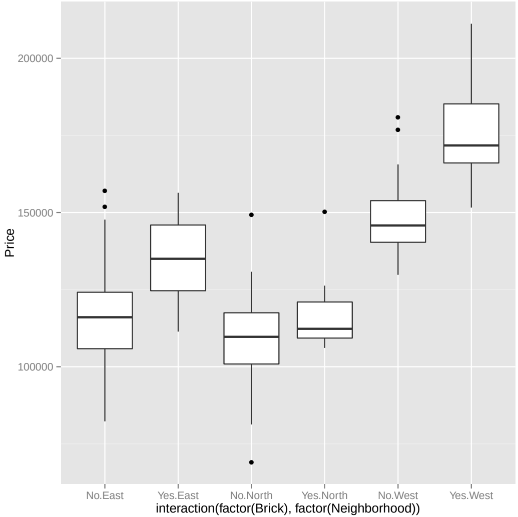
Grouped Boxplots (Several Variables)

To see the effect of several variables X_1 and X_2 on a target Y jointly, one can group data by their **interaction**, i.e., pairs of values (x_1, x_2):

$$\pi_Y(\mathcal{D}^{\text{train}}|_{X_1=x_1}|_{X_2=x_2}), \quad \text{for } x_1 \in \text{dom}\, X_1, x_2 \in \text{dom}\, X_2$$

$$\text{with } \mathcal{D}|_{X=x} := \{(x', y) \in \mathcal{D} \mid x' = x\},$$
$$\pi_Y \mathcal{D} := \{y \in \text{dom}\, Y \mid (x, y) \in \mathcal{D}\}$$

Example: House Prices: Multiple Dependencies



Conditionally Constant Models (Several Variables)

One can build more fine-grained models by conditioning on several variables jointly.

- to optimize RMSE, one computes the group means:

$$\hat{y}(x) := \text{mean}\, \pi_Y(\mathcal{D}^{\text{train}}|_{X_1=x_1}|_{X_2=x_2})$$

- to optimize MAE, one computes the group medians:

$$\hat{y}(x) := \text{median}\, \pi_Y(\mathcal{D}^{\text{train}}|_{X_1=x_1}|_{X_2=x_2})$$

Example: House Prices: Conditionally Constant Models

X	RMSE($\mathcal{D}^{\text{train}}$)
—	28 035.64
Bedrooms	24 194.59
Bathrooms	—
Offers	27 401.65
Brick	24 350.58
Neighborhood	18 056.69
Brick $\times$ Neighborhood	16 565.43

Note: 50:50 train/test split. By Bathrooms fails due to empty cell in train.

Conditional Constant Models for Classification

For other target spaces and losses such as

- binary classification
- multiclass classification
- multi-label classification
- etc.

conditional constant models work the same way: to compute the group aggregates for each level of the grouping variable, i.e.,

- binary classification: majority label, relative class frequencies
- multiclass classification: majority label, relative class frequencies
- multi-label classification: relative class frequencies

Issues and Ideas

1. **empty cells**:
   - if there are no samples with a specific predictor value in train, one cannot learn a group aggregate.
   - fix: resort to total sample aggregate.

Issues and Ideas

1. **empty cells**:
   - if there are no samples with a specific predictor value in train, one cannot learn a group aggregate.
   - fix: resort to total sample aggregate.

2. **continuous predictors**:
   - for continuous predictors X there are no natural groups with the same value.
   - fix: discretize/bin the continuous predictor.
     - disadvantage: information about similarity between different levels is lost.
     - advantage: can capture non-linear effects.

Issues and Ideas

1. **empty cells**:
   - if there are no samples with a specific predictor value in train, one cannot learn a group aggregate.
   - fix: resort to total sample aggregate.

2. **continuous predictors**:
   - for continuous predictors X there are no natural groups with the same value.
   - fix: discretize/bin the continuous predictor.
     - disadvantage: information about similarity between different levels is lost.
     - advantage: can capture non-linear effects.

3. **low-frequency cells**:
   - if there are only a few samples with a specific predictor value in train, the group aggregate may not be learnt accurately.

Outlook

1. One does not have to compute all cells on a grid $\text{dom}\, X_1 \times \text{dom}\, X_2$, but one can choose different variables $X_2(x_1)$ to combine with different values of $x_1 \in X_1$ **(decision trees)**.
   - conditional constant models on a single predictor can be interpreted as **decision tree stumps**.

Outlook

1. One does not have to compute all cells on a grid $\operatorname{dom} X_1 \times \operatorname{dom} X_2$, but one can choose different variables $X_2(x_1)$ to combine with different values of $x_1 \in X_1$ **(decision trees)**.
   - conditional constant models on a single predictor can be interpreted as **decision tree stumps**.

2. One can represent conditional constant models by **linear models** (see section 1.3) on indicator variables for nominal levels.

Outlook

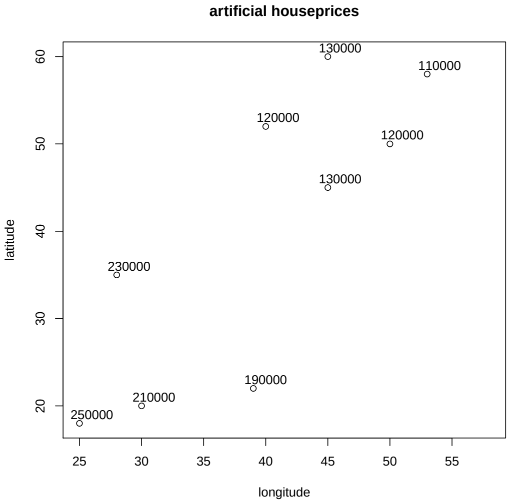
1. One does not have to compute all cells on a grid $\operatorname{dom} X_1 \times \operatorname{dom} X_2$, but one can choose different variables $X_2(x_1)$ to combine with different values of $x_1 \in X_1$ **(decision trees)**.
   - conditional constant models on a single predictor can be interpreted as **decision tree stumps**.

2. One can represent conditional constant models by **linear models** (see section 1.3) on indicator variables for nominal levels.

3. To capture interactions between nominal predictors with many levels, one can use **factorization models** (see chapter 5).

Outline

Idea


artificial houseprices
latitude
longitude
60
50
40
30
20
25
30
35
40
45
50
55
130000
110000
120000
120000
130000
230000
190000
210000
250000

Idea

A nearest neighbor model predicts for an instance $x \in \mathcal{X}$
the **aggregate** of the target values y'
of the nearest neighbors $(x', y') \in \mathcal{D}^{\text{train}}$,
i.e., of the training instances with **smallest distance** $d(x, x')$.

Idea

A nearest neighbor model predicts for an instance $x \in \mathcal{X}$
the **aggregate** of the target values y'
of the nearest neighbors $(x', y') \in \mathcal{D}^{\text{train}}$,
i.e., of the training instances with **smallest distance** $d(x, x')$.

Distance measures:

- function $d : \mathcal{X} \times \mathcal{X} \to \mathbb{R}_0^+$, e.g., for $\mathcal{X} := \mathbb{R}^m$
- Euclidean distance / L_2 distance:

$$d(x, x') := \sqrt{\sum_{i=1}^{m} (x_i - x'_i)^2}$$

- Manhattan distance / L_1 distance:

$$d(x, x') := \sum_{i=1}^{m} |x_i - x'_i|$$

Idea

A nearest neighbor model predicts for an instance $x \in \mathcal{X}$
the **aggregate** of the target values y'
of the nearest neighbors $(x', y') \in \mathcal{D}^{\text{train}}$,
i.e., of the training instances with **smallest distance** $d(x, x')$.

How many neighbors?

- fix a number $k \in \mathbb{N}$ of nearest neighbors to select.

Idea

A nearest neighbor model predicts for an instance $x \in \mathcal{X}$
the **aggregate** of the target values y'
of the nearest neighbors $(x', y') \in \mathcal{D}^{\text{train}}$,
i.e., of the training instances with **smallest distance** $d(x, x')$.

Aggregate:

- continuous target, RMSE loss: average.
- continuous target, MAE loss: median.

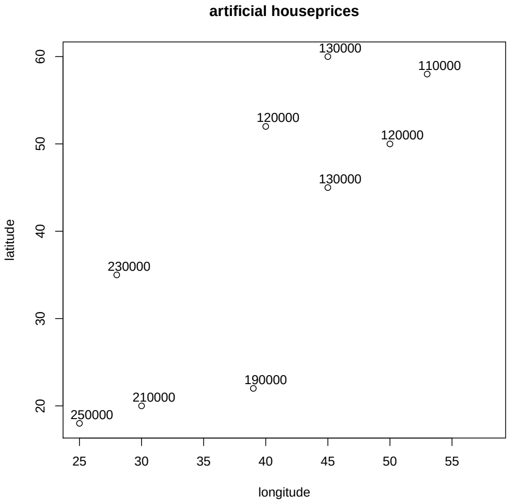
  $\vdots$
- nominal target, misclassification rate: majority class.
- nominal target, squared loss: relative class frequencies.

  $\vdots$

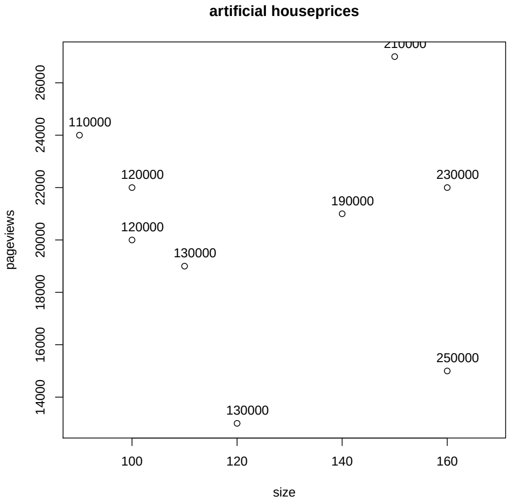
Example: Artificial Houseprice Data

	longitude	latitude	size	pageviews	price
1	50	50	100	22000	120000
2	45	60	120	13000	130000
3	53	58	90	24000	110000
4	40	52	100	20000	120000
5	45	45	110	19000	130000
6	30	20	150	27000	210000
7	39	22	140	21000	190000
8	25	18	160	15000	250000
9	28	35	160	22000	230000

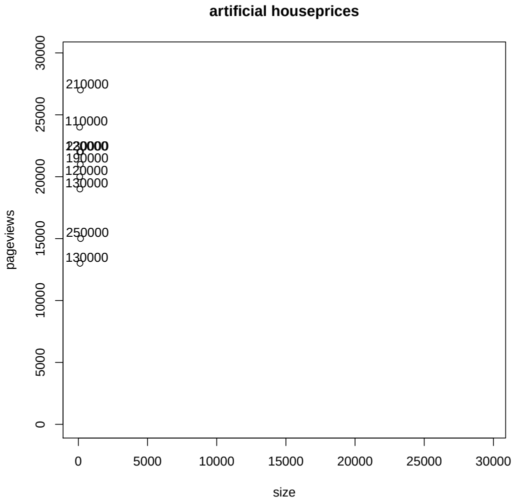
Example: Artificial Houseprice Data



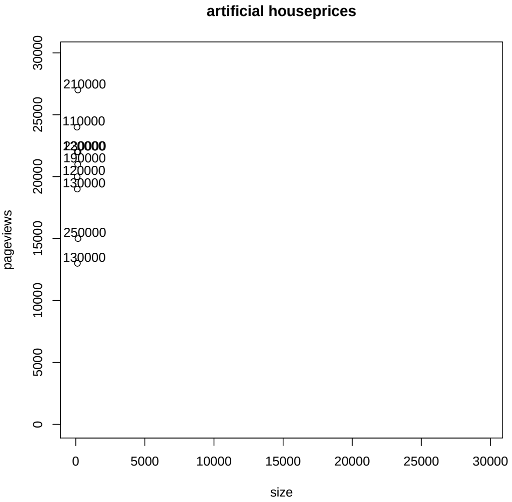
Example: Artificial Houseprice Data



Example: Artificial Houseprice Data



Example: Artificial Houseprice Data



Note: This is the data as seen by the Euclidean distance without normalization!

Prediction Formula

$$\hat{y}(x) := \text{aggregate}(N_k(\mathcal{D}^{\text{train}}, x))$$

where $$N_k(\mathcal{D}, x) := \underset{(x',y')\in\mathcal{D}}{\overset{k}{\arg\min}}\; d(x, x') \quad (\textbf{neighborhood})$$

i.e., for continuous targets and RMSE loss

$$\hat{y}(x) := \text{mean}(\pi_Y(N_k(\mathcal{D}^{\text{train}}, x))) = \frac{1}{k} \sum_{(x',y')\in N_k(\mathcal{D}^{\text{train}},x)} y'$$

and for nominal targets and squared loss

$$\hat{p}(Y = y|x) := \frac{1}{k} |\{(x', y') \in N_k(\mathcal{D}^{\text{train}}, x) \mid y = y'\}|$$

Inference Algorithm

To compute k-nearest neighbors in a naive way, for every query $x \in X$ the whole training set $\mathcal{D}^{\text{train}}$ can be sorted by increasing distance $d(x, \cdot)$ to the query instance

$$\mathcal{D}^{\text{train}} = \{(x_{(1)}, y_{(1)}), (x_{(2)}, y_{(2)}), (x_{(3)}, y_{(3)}), \ldots, (x_{(n)}, y_{(n)})\}$$
$$\text{with } d(x, x_{(1)}) \leq d(x, x_{(2)}) \leq d(x, x_{(3)}) \leq \ldots d(x, x_{(n)})$$

and then the first k instances be taken:

$$N_k(\mathcal{D}^{\text{train}}, x) = \{(x_{(1)}, y_{(1)}), (x_{(2)}, y_{(2)}), (x_{(3)}, y_{(3)}), \ldots, (x_{(k)}, y_{(k)})\}$$

Note: Instead of a full sort with complexity $O(n \log n)$, a partial sorting such as partial quicksort with complexity $O(n + k \log k)$ [Ano13] should be used; or a naive online selection with complexity $O(nk)$.

Outline

Bayes' Rule

For two random variables X, Y:

$$p(Y \mid X) = \frac{p(X \mid Y)\, p(Y)}{p(X)}$$

$$p(X = x) = \sum_{y' \in \mathrm{dom}\, Y} p(X = x \mid Y = y')\, p(Y = y')$$

Bayes' Rule / Example: start-ups

$$\text{dom}\, Y := \{\text{success}, \text{failure}\}, \text{dom}\, X := \{\text{plan}, \text{no plan}\}$$

$$p(X = \text{plan} \mid Y = \text{succ}) = \frac{9}{10}, \quad p(X = \text{plan} \mid Y = \text{fail}) = \frac{1}{2}$$

$$p(Y = \text{succ}) = \frac{1}{20} = 0.05$$

Bayes' Rule / Example: start-ups

$$\text{dom } Y := \{\text{success}, \text{failure}\}, \text{dom } X := \{\text{plan}, \text{no plan}\}$$

$$p(X = \text{plan} \mid Y = \text{succ}) = \frac{9}{10}, \quad p(X = \text{plan} \mid Y = \text{fail}) = \frac{1}{2}$$

$$p(Y = \text{succ}) = \frac{1}{20} = 0.05$$

$$p(\text{succ} \mid \text{plan}) = \frac{p(\text{plan} \mid \text{succ})\, p(\text{succ})}{p(\text{plan} \mid \text{succ})\, p(\text{succ}) + p(\text{plan} \mid \text{fail})\, p(\text{fail})}$$

Bayes' Rule / Example: start-ups

$$\text{dom } Y := \{\text{success}, \text{failure}\}, \text{dom } X := \{\text{plan}, \text{no plan}\}$$

$$p(X = \text{plan} \mid Y = \text{succ}) = \frac{9}{10}, \quad p(X = \text{plan} \mid Y = \text{fail}) = \frac{1}{2}$$

$$p(Y = \text{succ}) = \frac{1}{20} = 0.05$$

$$p(\text{succ} \mid \text{plan}) = \frac{p(\text{plan} \mid \text{succ})\, p(\text{succ})}{p(\text{plan} \mid \text{succ})\, p(\text{succ}) + p(\text{plan} \mid \text{fail})\, p(\text{fail})}$$

$$= \frac{\frac{9}{10} \cdot \frac{1}{20}}{\frac{9}{10} \cdot \frac{1}{20} + \frac{1}{2} \cdot \frac{19}{20}} = 0.087$$

Bayes' Rule / Example: start-ups

$$\text{dom } Y := \{\text{success}, \text{failure}\}, \text{dom } X := \{\text{plan}, \text{no plan}\}$$

$$p(X = \text{plan} \mid Y = \text{succ}) = \frac{9}{10}, \quad p(X = \text{plan} \mid Y = \text{fail}) = \frac{1}{2}$$

$$p(Y = \text{succ}) = \frac{1}{20} = 0.05$$

$$p(\text{succ} \mid \text{plan}) = \frac{p(\text{plan} \mid \text{succ})\, p(\text{succ})}{p(\text{plan} \mid \text{succ})\, p(\text{succ}) + p(\text{plan} \mid \text{fail})\, p(\text{fail})}$$

$$= \frac{\frac{9}{10} \cdot \frac{1}{20}}{\frac{9}{10} \cdot \frac{1}{20} + \frac{1}{2} \cdot \frac{19}{20}} = 0.087$$

$$p(\text{succ} \mid \text{no plan}) = \frac{p(\text{no plan} \mid \text{succ})\, p(\text{succ})}{p(\text{no plan} \mid \text{succ})\, p(\text{succ}) + p(\text{no plan} \mid \text{fail})\, p(\text{fail})}$$

$$= \frac{\frac{1}{10} \cdot \frac{1}{20}}{\frac{1}{10} \cdot \frac{1}{20} + \frac{1}{2} \cdot \frac{19}{20}} = 0.0104$$

Estimate Probabilities from Data

no.	X	Y
1	plan	fail
⋮	⋮	⋮
95	plan	fail
96	no plan	fail
⋮	⋮	⋮
190	no plan	fail
191	plan	succ
⋮	⋮	⋮
199	plan	succ
200	no plan	succ

Bayes' Rule for Prediction

If used for predicting Y, the denominator in Bayes' rule can be omitted:

$$p(Y \mid X) = \frac{p(X \mid Y)\, p(Y)}{p(X)}$$
$$\propto p(X \mid Y)\, p(Y)$$

Bayes' Rule for Prediction

If used for predicting Y, the denominator in Bayes' rule can be omitted:

$$p(Y \mid X) = \frac{p(X \mid Y)\, p(Y)}{p(X)}$$
$$\propto p(X \mid Y)\, p(Y)$$

Example:

$$p(\text{succ} \mid \text{plan}) \propto p(\text{plan} \mid \text{succ})\, p(\text{succ}) \qquad = \frac{9}{10} \cdot \frac{1}{20} = 0.045$$
$$p(\text{fail} \mid \text{plan}) \propto p(\text{plan} \mid \text{fail})\, p(\text{fail}) \qquad = \frac{1}{2} \cdot \frac{19}{20} = 0.0475$$

$\rightsquigarrow$ failure is more likely, even with a business plan.

Multiple Predictors / Naive Bayes Assumption

For multiple predictors $X_1, X_2, \ldots, X_p$,

$$p(X_1, X_2, \ldots, X_p \mid Y)$$

usually is hard to estimate.

The **Naive Bayes** model **assumes** that **all the predictors are independent given the target**:

$$p(X_1, X_2, \ldots, X_p \mid Y) = p(X_1 \mid Y)\, p(X_2 \mid Y) \cdots p(X_p \mid Y)$$

Example

Artificial data about visitors of an online shop:

	referrer	num.visits	duration	buyer
1	search engine	several	15	yes
2	search engine	once	10	yes
3	other	several	5	yes
4	ad	once	15	yes
5	ad	once	10	no
6	other	once	10	no
7	other	once	5	no
8	ad	once	5	no

Example

Artificial data about visitors of an online shop:

	referrer	num.visits	duration	buyer
1	search engine	several	15	yes
2	search engine	once	10	yes
3	other	several	5	yes
4	ad	once	15	yes
5	ad	once	10	no
6	other	once	10	no
7	other	once	5	no
8	ad	once	5	no

$$p(Y = \text{yes}) = 0.5$$

Example

Artificial data about visitors of an online shop:

	referrer	num.visits	duration	buyer
1	search engine	several	15	yes
2	search engine	once	10	yes
3	other	several	5	yes
4	ad	once	15	yes
5	ad	once	10	no
6	other	once	10	no
7	other	once	5	no
8	ad	once	5	no

$$p(X_1 = \text{search} \mid Y = \text{yes}) = 0.5 \qquad p(X_1 = \text{search} \mid Y = \text{no}) = 0.0$$
$$p(X_1 = \text{ad} \mid Y = \text{yes}) = 0.25 \qquad p(X_1 = \text{ad} \mid Y = \text{no}) = 0.5$$
$$p(X_1 = \text{other} \mid Y = \text{yes}) = 0.25 \qquad p(X_1 = \text{other} \mid Y = \text{no}) = 0.5$$

Example

Artificial data about visitors of an online shop:

	referrer	num.visits	duration	buyer
1	search engine	several	15	yes
2	search engine	once	10	yes
3	other	several	5	yes
4	ad	once	15	yes
5	ad	once	10	no
6	other	once	10	no
7	other	once	5	no
8	ad	once	5	no

$$p(X_2 = \text{several} \mid Y = \text{yes}) = 0.5 \qquad p(X_2 = \text{several} \mid Y = \text{no}) = 0.0$$
$$p(X_2 = \text{once} \mid Y = \text{yes}) = 0.5 \qquad p(X_2 = \text{once} \mid Y = \text{no}) = 1.0$$

Example

Artificial data about visitors of an online shop:

	referrer	num.visits	duration	buyer
1	search engine	several	15	yes
2	search engine	once	10	yes
3	other	several	5	yes
4	ad	once	15	yes
5	ad	once	10	no
6	other	once	10	no
7	other	once	5	no
8	ad	once	5	no

$$p(X_3 = 5 \mid Y = \text{yes}) = 0.25 \qquad p(X_3 = 5 \mid Y = \text{no}) = 0.5$$
$$p(X_3 = 10 \mid Y = \text{yes}) = 0.25 \qquad p(X_3 = 10 \mid Y = \text{no}) = 0.5$$
$$p(X_3 = 15 \mid Y = \text{yes}) = 0.5 \qquad p(X_3 = 15 \mid Y = \text{no}) = 0.0$$

Example / Model Parameters

$$p(Y = \text{yes}) = 0.5$$

$$
\begin{aligned}
p(X_1 = \text{search} \mid Y = \text{yes}) &= 0.5 & p(X_1 = \text{search} \mid Y = \text{no}) &= 0.0 \\
p(X_1 = \text{ad} \mid Y = \text{yes}) &= 0.25 & p(X_1 = \text{ad} \mid Y = \text{no}) &= 0.5 \\
p(X_1 = \text{other} \mid Y = \text{yes}) &= 0.25 & p(X_1 = \text{other} \mid Y = \text{no}) &= 0.5 \\
p(X_2 = \text{several} \mid Y = \text{yes}) &= 0.5 & p(X_2 = \text{several} \mid Y = \text{no}) &= 0.0 \\
p(X_2 = \text{once} \mid Y = \text{yes}) &= 0.5 & p(X_2 = \text{once} \mid Y = \text{no}) &= 1.0 \\
p(X_3 = 5 \mid Y = \text{yes}) &= 0.25 & p(X_3 = 5 \mid Y = \text{no}) &= 0.5 \\
p(X_3 = 10 \mid Y = \text{yes}) &= 0.25 & p(X_3 = 10 \mid Y = \text{no}) &= 0.5 \\
p(X_3 = 15 \mid Y = \text{yes}) &= 0.5 & p(X_3 = 15 \mid Y = \text{no}) &= 0.0
\end{aligned}
$$

Will a visitor with X_1 =ad, X_2=once, X_3=10 buy?

Example / Model Parameters

$$p(Y = \text{yes}) = 0.5$$

$$p(X_1 = \text{search} \mid Y = \text{yes}) = 0.5 \qquad p(X_1 = \text{search} \mid Y = \text{no}) = 0.0$$
$$p(X_1 = \text{ad} \mid Y = \text{yes}) = 0.25 \qquad p(X_1 = \text{ad} \mid Y = \text{no}) = 0.5$$
$$p(X_1 = \text{other} \mid Y = \text{yes}) = 0.25 \qquad p(X_1 = \text{other} \mid Y = \text{no}) = 0.5$$
$$p(X_2 = \text{several} \mid Y = \text{yes}) = 0.5 \qquad p(X_2 = \text{several} \mid Y = \text{no}) = 0.0$$
$$p(X_2 = \text{once} \mid Y = \text{yes}) = 0.5 \qquad p(X_2 = \text{once} \mid Y = \text{no}) = 1.0$$
$$p(X_3 = 5 \mid Y = \text{yes}) = 0.25 \qquad p(X_3 = 5 \mid Y = \text{no}) = 0.5$$
$$p(X_3 = 10 \mid Y = \text{yes}) = 0.25 \qquad p(X_3 = 10 \mid Y = \text{no}) = 0.5$$
$$p(X_3 = 15 \mid Y = \text{yes}) = 0.5 \qquad p(X_3 = 15 \mid Y = \text{no}) = 0.0$$

Will a visitor with X_1 =ad, X_2=once, X_3=10 buy?

$$\begin{aligned}
q_{\text{yes}} &= q(Y = \text{yes} \mid X_1 = \text{ad}, X_2 = \text{once}, X_3 = 10) \\
&= p(Y = \text{yes})\, p(X_1 = \text{ad} \mid Y = \text{yes}) \\
&\qquad p(X_2 = \text{once} \mid Y = \text{yes})\, p(X_3 = 10) \mid Y = \text{yes}) \\
&= 0.5 \cdot 0.25 \cdot 0.5 \cdot 0.25 = 0.015625
\end{aligned}$$

Example / Model Parameters

$$p(Y = \text{yes}) = 0.5$$

$$
\begin{aligned}
p(X_1 = \text{search} \mid Y = \text{yes}) &= 0.5 & p(X_1 = \text{search} \mid Y = \text{no}) &= 0.0\\
p(X_1 = \text{ad} \mid Y = \text{yes}) &= 0.25 & p(X_1 = \text{ad} \mid Y = \text{no}) &= 0.5\\
p(X_1 = \text{other} \mid Y = \text{yes}) &= 0.25 & p(X_1 = \text{other} \mid Y = \text{no}) &= 0.5\\
p(X_2 = \text{several} \mid Y = \text{yes}) &= 0.5 & p(X_2 = \text{several} \mid Y = \text{no}) &= 0.0\\
p(X_2 = \text{once} \mid Y = \text{yes}) &= 0.5 & p(X_2 = \text{once} \mid Y = \text{no}) &= 1.0\\
p(X_3 = 5 \mid Y = \text{yes}) &= 0.25 & p(X_3 = 5 \mid Y = \text{no}) &= 0.5\\
p(X_3 = 10 \mid Y = \text{yes}) &= 0.25 & p(X_3 = 10 \mid Y = \text{no}) &= 0.5\\
p(X_3 = 15 \mid Y = \text{yes}) &= 0.5 & p(X_3 = 15 \mid Y = \text{no}) &= 0.0
\end{aligned}
$$

Will a visitor with X_1 =ad, X_2=once, X_3=10 buy?

$$
\begin{aligned}
q_{\text{no}} &= q(Y = \text{no} \mid X_1 = \text{search}, X_2 = \text{once}, X_3 = 10)\\
&= p(Y = \text{no})\, p(X_1 = \text{ad} \mid Y = \text{no})\\
&\qquad p(X_2 = \text{once} \mid Y = \text{no})\, p(X_3 = 10) \mid Y = \text{no})\\
&= 0.5 \cdot 0.5 \cdot 1.0 \cdot 0.5 = 0.125
\end{aligned}
$$

Example / Model Parameters

$$p(Y = \text{yes}) = 0.5$$

$p(X_1 = \text{search} \mid Y = \text{yes}) = 0.5$	$p(X_1 = \text{search} \mid Y = \text{no}) = 0.0$
$p(X_1 = \text{ad} \mid Y = \text{yes}) = 0.25$	$p(X_1 = \text{ad} \mid Y = \text{no}) = 0.5$
$p(X_1 = \text{other} \mid Y = \text{yes}) = 0.25$	$p(X_1 = \text{other} \mid Y = \text{no}) = 0.5$
$p(X_2 = \text{several} \mid Y = \text{yes}) = 0.5$	$p(X_2 = \text{several} \mid Y = \text{no}) = 0.0$
$p(X_2 = \text{once} \mid Y = \text{yes}) = 0.5$	$p(X_2 = \text{once} \mid Y = \text{no}) = 1.0$
$p(X_3 = 5 \mid Y = \text{yes}) = 0.25$	$p(X_3 = 5 \mid Y = \text{no}) = 0.5$
$p(X_3 = 10 \mid Y = \text{yes}) = 0.25$	$p(X_3 = 10 \mid Y = \text{no}) = 0.5$
$p(X_3 = 15 \mid Y = \text{yes}) = 0.5$	$p(X_3 = 15 \mid Y = \text{no}) = 0.0$

Will a visitor with X_1 =ad, X_2=once, X_3=10 buy?

$$p(Y = \text{yes} \mid X_1 = \text{ad}, X_2 = \text{once}, X_3 = 10) = \frac{q_{\text{no}}}{q_{\text{no}} + q_{\text{yes}}} = \frac{0.015625}{0.015625 + 0.125} = 0.111$$

Learning Algorithm

Given training data $\mathcal{D}^{\text{train}}$, compute

$$\alpha_y := \hat{p}(Y = y) := \frac{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}{|\mathcal{D}^{\text{train}}|}$$

$$\beta_{y,i,x} := \hat{p}(X_i = x \mid Y = y) := \frac{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y, x'_i = x\}|}{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}$$

for $y \in \text{dom}\, Y, x \in \text{dom}\, X_i, i = 1, \ldots, p$.

Learning Algorithm

Given training data $\mathcal{D}^{\text{train}}$, compute

$$\alpha_y := \hat{p}(Y = y) := \frac{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}{|\mathcal{D}^{\text{train}}|}$$

$$\beta_{y,i,x} := \hat{p}(X_i = x \mid Y = y) := \frac{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y, x'_i = x\}|}{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}$$

for $y \in \operatorname{dom} Y, x \in \operatorname{dom} X_i, i = 1, \ldots, p$.

For nominal predictor variables with rare levels, usually a **Laplace smoothing** of size $n_0 \in \mathbb{R}_0^+$ is added:

$$\beta_{y,i,x} := \hat{p}(X_i = x \mid Y = y) := \frac{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y, x' = x\}| + n_0}{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}| + n_0 |\operatorname{dom} X_i|}$$

Inference Algorithm

Given $x := (x_1, x_2, \ldots, x_p) \in \text{dom}\, X$, i.e., $x_i \in \text{dom}\, X_i, i = 1, \ldots, p$, compute

$$\hat{p}(Y = y \mid X_1 = x_1, \ldots, X_p = x_p)$$
$$= \frac{\hat{p}(Y = y) \prod_{i=1}^{p} \hat{p}(X_i = x_i \mid Y = y)}{\sum_{y' \in \text{dom}\, Y} \hat{p}(Y = y') \prod_{i=1}^{p} \hat{p}(X_i = x_i \mid Y = y')}$$

for all $y \in \text{dom}\, Y$.

Inference Algorithm

Given $x := (x_1, x_2, \ldots, x_p) \in \text{dom}\, X$, i.e., $x_i \in \text{dom}\, X_i, i = 1, \ldots, p$, compute

$$\begin{aligned}\hat{p}(Y = y \,|X_1 = x_1, \ldots, X_p = x_p) \\ &= \frac{\hat{p}(Y = y) \prod_{i=1}^{p} \hat{p}(X_i = x_i \mid Y = y)}{\sum_{y' \in \text{dom}\, Y} \hat{p}(Y = y') \prod_{i=1}^{p} \hat{p}(X_i = x_i \mid Y = y')} \\ &= \frac{\alpha_y \prod_{i=1}^{p} \beta_{y,i,x_i}}{\sum_{y' \in \text{dom}\, Y} \alpha_{y'} \prod_{i=1}^{p} \beta_{y',i,x_i}}\end{aligned}$$

for all $y \in \text{dom}\, Y$.

Inference Algorithm

Given $x := (x_1, x_2, \ldots, x_p) \in \operatorname{dom} X$, i.e., $x_i \in \operatorname{dom} X_i$, $i = 1, \ldots, p$, compute

$$
\begin{aligned}
\hat{p}(Y = y \,|X_1 = x_1, \ldots, X_p = x_p) \\
&= \frac{\hat{p}(Y = y) \prod_{i=1}^{p} \hat{p}(X_i = x_i \mid Y = y)}{\sum_{y' \in \operatorname{dom} Y} \hat{p}(Y = y') \prod_{i=1}^{p} \hat{p}(X_i = x_i \mid Y = y')} \\
&= \frac{\alpha_y \prod_{i=1}^{p} \beta_{y,i,x_i}}{\sum_{y' \in \operatorname{dom} Y} \alpha_{y'} \prod_{i=1}^{p} \beta_{y',i,x_i}}
\end{aligned}
$$

for all $y \in \operatorname{dom} Y$.

Computed via

$$q_y := \alpha_y \prod_{i=1}^{p} \beta_{y,i,x_i}$$

$$Q := \sum_{y \in \operatorname{dom} Y} q_y$$

$$p(Y = y \,|X_1 = x_1, \ldots, X_p = x_p) = \frac{q_y}{Q}$$

Continuous Predictors

For nominal predictors X_i, we can simply estimate

$$p(X_i = x \mid Y = y)$$

for **all possible values** x by their (smoothed) relative frequency within instances with target y (categorical distribution).

For a continuous predictor X_i this will not work.

Continuous Predictors: Gaussian Naive Bayes

But we can replace the categorical distribution by any other distribution for X_i, e.g., a Gaussian distribution

$$p(X_i = x \mid Y = y) = \frac{1}{\sqrt{2\pi\sigma^2_{y,i}}} e^{\frac{-(x-\mu_{y,i})^2}{2\sigma^2_{y,i}}}, \mu_{y,i}, \sigma^2_{y,i} \in \mathbb{R}$$

Their parameters $\mu_{y,i}, \sigma^2_{y,i}$ have to be learned from data:

$$\mu_{y,i} := \text{mean}\, \pi_{X_i}(\mathcal{D}^{\text{train}}|_{Y=y}) = \frac{\sum_{(x',y')\in\mathcal{D}^{\text{train}}, y'=y} x'_i}{|\{(x',y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}$$

$$\sigma^2_{y,i} := \text{variance}\, \pi_{X_i}(\mathcal{D}^{\text{train}}|_{Y=y}) = \frac{\sum_{(x',y')\in\mathcal{D}^{\text{train}}, y'=y} (x'_i - \mu_{y,i})^2}{|\{(x',y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}$$

Which Loss does Naive Bayes Minimize?

Naive Bayes maximizes the likelihood (= minimizes the negative log-likelihood):

$$\begin{aligned}\ell(\mathcal{D}, \hat{p}) &= \prod_{(x,y)\in\mathcal{D}} \hat{p}(X = x, Y = y)\\ &= \prod_{(x,y)\in\mathcal{D}} \hat{p}(Y = y)\hat{p}(X = x \mid Y = y)\end{aligned}$$

Proof.

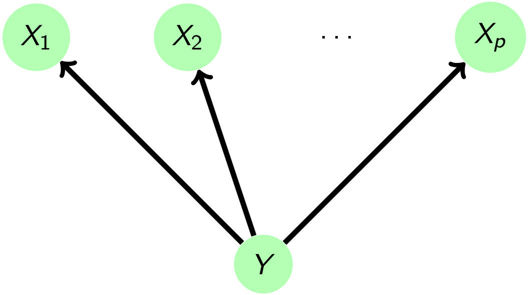
$$\ell(\mathcal{D}, \hat{p}) = \prod_{y\in\mathrm{dom}\, Y} \hat{p}(Y = y)^{n_y} \prod_{i=1}^{p} \prod_{x\in\mathrm{dom}\, X_i} \hat{p}(X_i = x \mid Y = y)^{n_{y,i,x}}$$

which according to lemma 1 assumes its minimum for

$$\hat{p}(Y = y) = \frac{n_y}{\sum_{y'\in\mathrm{dom}\, Y} n_{y'}}, \quad \hat{p}(X_i = x \mid Y = y) = \frac{n_{y,i,x}}{\sum_{x'\in\mathrm{dom}\, X_i} n_{y,i,}}$$

with $n_y := |\{(x', y') \in \mathcal{D} \mid y' = y\}|, \quad n_{y,i,x} := |\{(x', y') \in \mathcal{D} \mid y' = y, x'_i = x$

A Simple Bayesian Network



$$p(\mathcal{V}) = \prod_{V \in \mathcal{V}} p(V \mid \text{parents of } V)$$

$$\overset{NB}{=} p(Y) \prod_{i=1}^{p} p(X_i \mid Y), \quad \mathcal{V} := \{Y, X_1, \ldots, X_p\}$$

Outline

Assumptions

Linear Discriminant Analysis (LDA) relies on the same decomposition

$$p(Y \mid X_1, \ldots, X_p) \propto p(X_1, \ldots, X_p \mid Y)\, p(Y)$$

as Naive Bayes, but does not assume that all the X_i are independent, but are **multivariate normal distributed**

$$p(X = x \mid Y = y) = \frac{1}{\sqrt{(2\pi)^p |\Sigma|}} e^{-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1} (x-\mu_y)}$$

with

- target-specific means $\mu_y \in \mathbb{R}^p$ and a
- **shared covariance matrix** $\Sigma \in \mathbb{R}^{p \times p}$.

Note: $x = (x_1, \ldots, x_p) \in \mathbb{R}^p$.

Learning Algorithm

Given training data $\mathcal{D}^{\text{train}}$, compute

$$\alpha_y := \hat{p}(Y = y) := \frac{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}{|\mathcal{D}^{\text{train}}|}$$

$$\begin{aligned}
\mu_y &:= \text{mean}\, \pi_X(\mathcal{D}^{\text{train}}|_{Y=y}) \\
&= \frac{\sum_{(x',y') \in \mathcal{D}^{\text{train}}, y'=y} x'}{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|} \\
\Sigma &:= \frac{1}{|\mathcal{D}^{\text{train}}|} \sum_{y \in \text{dom}\, Y} |(\mathcal{D}^{\text{train}}|_{Y=y})| \,\text{cov}\, \pi_X(\mathcal{D}^{\text{train}}|_{Y=y}) \\
&= \frac{\sum_{(x',y') \in \mathcal{D}^{\text{train}}} (x' - \mu_{y'})^T (x' - \mu_{y'})}{|\mathcal{D}^{\text{train}}|}
\end{aligned}$$

for $y \in \text{dom}\, Y$.

Inference Algorithm

Given $x \in \operatorname{dom} X := \mathbb{R}^p$, compute

$$\begin{aligned}
\hat{p}(Y = y \mid X = x) \propto & \hat{p}(X = x \mid Y = y)\, \hat{p}(Y = y) \\
= & \frac{1}{\sqrt{(2\pi)^p |\Sigma|}} e^{-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1} (x-\mu_y)} \alpha_y \\
\propto & e^{-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1} (x-\mu_y)} \alpha_y
\end{aligned}$$

for all $y \in \operatorname{dom} Y$.

Inference Algorithm

Given $x \in \operatorname{dom} X := \mathbb{R}^p$, compute

$$\begin{aligned}
\hat{p}(Y = y \mid X = x) &\propto \hat{p}(X = x \mid Y = y)\, \hat{p}(Y = y) \\
&= \frac{1}{\sqrt{(2\pi)^p |\Sigma|}} e^{-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1} (x-\mu_y)} \alpha_y \\
&\propto e^{-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1} (x-\mu_y)} \alpha_y
\end{aligned}$$

for all $y \in \operatorname{dom} Y$.

Computed via

$$q_y := e^{-\frac{1}{2}(x-\mu_y)^T \Sigma^{-1} (x-\mu_y)} \alpha_y$$

$$Q := \sum_{y \in \operatorname{dom} Y} q_y$$

$$\hat{p}(Y = y \mid X = x) = \frac{q_y}{Q}$$

LDA, QDA and Gaussian Naive Bayes

Quadratic Discriminant Analysis (QDA):

- the covariance matrix Σ also is target specific:

$$p(X = x \mid Y = y) = \frac{1}{\sqrt{(2\pi)^p |\Sigma_{\mathbf{y}}|}} e^{-\frac{1}{2}(x-\mu_y)^T \Sigma_{\mathbf{y}}^{-1}(x-\mu_y)}$$

- its estimation is simply

$$\Sigma_y := \operatorname{cov} \pi_X(\mathcal{D}^{\text{train}}|_{Y=y}) = \frac{\sum_{(x',y')\in\mathcal{D}^{\text{train}}, y'=y} (x' - \mu_{y'})^T (x' - \mu_{y'})}{|\{(x', y') \in \mathcal{D}^{\text{train}} \mid y' = y\}|}$$

- QDA requires $|\operatorname{dom} Y|$ as many parameters to be estimated for the covariance matrices compared to LDA, and the full covariance matrix requires already $\frac{p(p+1)}{2}$ parameters.

QDA and Gaussian Naive Bayes

The **Gaussian Naive Bayes** model is a special case of a QDA with diagonal covariance matrices:

$$\Sigma_y = \begin{pmatrix} \sigma^2_{y,1} & 0 & \dots & \dots & 0 \\ 0 & \sigma^2_{y,2} & 0 & \dots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & \sigma^2_{y,p-1} & 0 \\ 0 & \dots & \dots & 0 & \sigma^2_{y,p} \end{pmatrix}$$

Outline

Which Model to Use?

Now we know different prediction models, e.g.,

- constant model,
- conditionally constant model,
- nearest neighbor model,
- Naive Bayes model,
- Linear Discriminant Analysis model.

Which one should we use for a specific task?

Which Model to Use?

Now we know different prediction models, e.g.,

- constant model,
- conditionally constant model,
- nearest neighbor model,
- Naive Bayes model,
- Linear Discriminant Analysis model.

Which one should we use for a specific task?

- depends on the task at hand.

Which Model to Use?

Now we know different prediction models, e.g.,

- constant model,
- conditionally constant model,
- nearest neighbor model,
- Naive Bayes model,
- Linear Discriminant Analysis model.

Which one should we use for a specific task?

- depends on the task at hand.

How can we find out?

- perform **model selection**.

Model Selection

We could use every model with 1/5 of our customers, and then we will see which was best.

Model Selection

We could use every model with 1/5 of our customers, and then we will see which was best.

- but we would like to know before we use the models for the real application.

Model Selection

We could use every model with 1/5 of our customers, and then we will see which was best.

- but we would like to know before we use the models for the real application.

We could do **as if** we would use the model, i.e.,

1. split the training data in **proper training data** and **validation data**,
2. train the models only on the proper training data,
3. evaluate the models on the validation data,
4. select the model that performs best on the validation data,
5. use this model for our application.

Model Selection

We could use every model with 1/5 of our customers, and then we will see which was best.

- but we would like to know before we use the models for the real application.

We could do **as if** we would use the model, i.e.,

1. split the training data in **proper training data** and **validation data**,
2. train the models only on the proper training data,
3. evaluate the models on the validation data,
4. select the model that performs best on the validation data,
5. **re-train the model on the whole training data.**
6. use this model for our application.

How to Split the Data?

Different types of splits are possible:

- 50% proper training, 50% validation
- 80% proper training, 20% validation
  - too few training data: training the models may be unreliable.
  - too few validation data: validation may be unreliable.

How to Split the Data?

Different types of splits are possible:

- 50% proper training, 50% validation
- 80% proper training, 20% validation
  - too few training data: training the models may be unreliable.
  - too few validation data: validation may be unreliable.

- ***n*-fold cross validation**:
  1. chop the data into n chunks of the same size,
  2. for $i = 1, \ldots, n$,
     - use all chunks but the i-th as proper training,
     - use the i-th chunk as validation
     - train and evaluate all models
  3. average all n evaluations

Which Model Configuration to Use?

Some models have different configurations:

- conditional constant model:
  - predictor variable X to condition on.
- nearest neighbor model:
  - neighborhood size k.
- Naive Bayes model:
  - Laplace smoothing n_0.
- Linear Discriminant Analysis:
  - [none]

often described by **hyperparameters**.

Which hyperparameter value to use?

Which Model Configuration to Use?

Some models have different configurations:

- conditional constant model:
  - predictor variable X to condition on.
- nearest neighbor model:
  - neighborhood size k.
- Naive Bayes model:
  - Laplace smoothing n_0.
- Linear Discriminant Analysis:
  - [none]

often described by **hyperparameters**.

Which hyperparameter value to use?

- depends on the data — use model selection to find out!

Which Predictor Variables to Use?

We can generically configure any model by selecting the predictor variables to use:

- use only the most-predictive one,
- use the 7 most predictive ones,
- use the 10% most predictive ones,
- use random 10 ones,
- use all,

  ⋮

Which predictor variables to use (**variable selection**)?

Which Predictor Variables to Use?

We can generically configure any model by selecting the predictor variables to use:

- use only the most-predictive one,
- use the 7 most predictive ones,
- use the 10% most predictive ones,
- use random 10 ones,
- use all,

  ⋮

Which predictor variables to use (**variable selection**)?

- depends on the data — use model selection to find out!

Which Predictor Variables to Use?

We can generically configure any model by selecting the predictor variables to use:

- use only the most-predictive one,
- use the 7 most predictive ones,
- use the 10% most predictive ones,
- use random 10 ones,
- use all,

  ⋮

Which predictor variables to use (**variable selection**)?

- depends on the data — use model selection to find out!?
- but there are 2^p many different such configurations !

Sequential Model Selection

To search for a good subset of predictor variables,
greedy stepwise removal (**backward search**) is applied:

1. start with all variables $V := \{1, 2, \ldots, p\}$.
2. train and evaluate the model on all variables.
3. for every variable $v \in V$:
   3.1 train and evaluate a model on variables $V \setminus \{v\}$.
4. if the best model $V \setminus \{v\}$ improves the model on V:
   4.1 $V := V \setminus \{v\}$
   4.2 go back to step 3.
5. return V

Note: The subset of predictor variables also can be interpreted as a (set-valued) hyperparameter.

Hyperparameter Interaction: **Grid Search**

Usually, hyperparameters interact, e.g.,

- using all variables, $k = 10$ neighbors may be optimal, but
- using only 10 variables, $k = 20$ neighbors may be optimal.

Hyperparameter Interaction: **Grid Search**

Usually, hyperparameters interact, e.g.,

- using all variables, $k = 10$ neighbors may be optimal, but
- using only 10 variables, $k = 20$ neighbors may be optimal.

Therefore,

- sequential model selection via sequential hyperparameter selection (select one hyperparameter at a time), generally is not save!
- Usually, all combinations of hyperparameter values have to be searched.

Hyperparameter Interaction: Grid Search

Usually, hyperparameters interact, e.g.,

- using all variables, $k = 10$ neighbors may be optimal, but
- using only 10 variables, $k = 20$ neighbors may be optimal.

Therefore,

- sequential model selection via sequential hyperparameter selection (select one hyperparameter at a time), generally is not save!
- Usually, all combinations of hyperparameter values have to be searched.

For numerical hyperparameters, one searches on a grid, e.g., for the neighborhood size:

1. coarse grid: $k = 1, \frac{1}{9}n, \frac{2}{9}n, \ldots, n$.
   - let k_0 be the best hyperparameter value on the coarse grid, k_{-1}, k_{+1} the one left and right of k_0.
2. finer grid: $k_{-1} + \frac{i}{11}(k_{+1} - k_{-1}),\ i = 1, \ldots, 10$.

Model Selection vs Model Combination: **Ensembles**

Instead of selecting the best model $\hat{y}$ out of a set of candidate models $\{\hat{y}_1, \ldots, \hat{y}_q\}$, one also can combine them, e.g.,

- **voting** (for nominal targets):
  choose the value most frequently predicted by member models

$$\hat{y}(x) := \operatorname*{arg\,max}_{y' \in \operatorname{dom} Y} \sum_{i=1}^{q} \delta(y' = \hat{y}_i(x))$$

- **averaging**:
  predict the mean of the values predicted by the member models

$$\hat{y}(x) := \operatorname{mean}\{\hat{y}_i(x) \mid i = 1, \ldots, q\} = \frac{1}{q} \sum_{i=1}^{q} \hat{y}_i(x)$$

Ensembles / Example

Binary classification: $p(Y = 1)$.

instance	NN	NB	LDA	voting	averaging
1	0.6	0.7	0.3	1.0	0.53
2	0.7	0.1	0.6	1.0	0.48
$\vdots$					

Model Selection vs Model Combination: **Ensembles** 2

Instead of selecting the best model $\hat{y}$ out of a set of candidate models $\{\hat{y}_1, \ldots, \hat{y}_q\}$, one also can combine them, e.g.,

- **weighted voting/averaging**:
  - let $w_i \in \mathbb{R}_0^+$ be a weight indicating the quality of model $\hat{y}_i$, e.g., its accuracy on a validation split,
  - predict the weighted mean of the values predicted by the member models

$$\hat{y}(x) := \frac{1}{\sum_{i=1}^q w_i} \sum_{i=1}^q w_i \hat{y}_i(x)$$

- **stacking**:
  - learn a **2nd stage model** $\hat{y}_{2\text{nd}}$ for the training data

$$\mathcal{D}_{2\text{nd}}^{\text{train}} := \{((\hat{y}_1(x), \ldots, \hat{y}_q(x)), y) \mid (x, y) \in \mathcal{D}^{\text{valid}}\}$$

  - predict the value of the 2nd stage model for the predictions of the member models:

$$\hat{y}(x) := \hat{y}_{2\text{nd}}(\hat{y}_1(x), \ldots, \hat{y}_q(x))$$

Outline

Summary Simple Models

	predictor	
target	continuous	nominal
continuous (regression)	Nearest Neighbor	Nearest Neighbor
nominal (classification)	Nearest Neighbor Naive Bayes / Gaussian LDA	Nearest Neighbor Naive Bayes (LDA)

Note: LDA (as any model for continuous predictors) can be used for nominal predictors after coding them as binary indicator variables.

Conclusion

- Prediction can be accomplished by several very simple models:
  - **Nearest Neighbor**: predicting the aggregated values of the closest training instances,
  - **Naive Bayes**: predicting the class that explains the predictors best, individually.
  - **Linear Discriminant Analysis**: predicting the class that explains the predictors best, collectively.

- Simple prediction models can be **trained by a single pass over the data**.

- These simple prediction models can be used to **get a first idea** about the scale of the prediction quality, i.e., how difficult a prediction problem is.

- These models usually **do not provide state-of-the-art prediction quality** and therefore should not be used in practice.

Readings

- Nearest Neighbor:
  - [HTFF05], ch. 13.3, [Bis06], ch. 2.5.2, [Mur12], ch. 1.4.2
- Naive Bayes:
  - [HTFF05], ch. 6.6.3, [Mur12], ch. 10.2.1
- LDA:
  - [HTFF05], ch. 4.3, [Bis06], ch. 4.1.4 and 4.1.6, [Mur12], ch. 4.2.2

References

Lemma
$f(x) = \prod_{i=1}^k x_i^{n_i}$ assumes its maximum on $\mathcal{X} := \{x \in \mathbb{R}^k \mid \sum_{i=1}^k x_i = 1\}$ at x^* with $x_i^* := \frac{n_i}{\sum_{i=1}^k n_i}$ $(i = 1, \ldots, k)$.

Proof.

$$g(x_1, \ldots, x_{k-1}) := \log f(x_1, \ldots, x_{k-1}, 1 - \sum_{i=1}^{k-1} x_i)$$

$$= n_k \log(1 - \sum_{i=1}^{k-1} x_i) + \sum_{i=1}^{k-1} n_i \log x_i$$

$$\frac{\partial g}{\partial x_i} = n_k \frac{1}{1 - \sum_{j=1}^{k-1} x_j}(-1) + n_i \frac{1}{x_i} \overset{!}{=} 0, \quad \forall i$$

$$\rightsquigarrow - n_k x_i + n_i(1 - \sum_{j=1}^{k-1} x_j) = 0, \quad \forall i \qquad (*)$$

References

Lemma

$f(x) = \prod_{i=1}^k x_i^{n_i}$ assumes its maximum on $\mathcal{X} := \{x \in \mathbb{R}^k \mid \sum_{i=1}^k x_i = 1\}$ at x^* with $x_i^* := \frac{n_i}{\sum_{i=1}^k n_i}$ $(i = 1, \dots, k)$.

Proof. (ctd.)

$$-n_k x_i + n_i(1 - \sum_{j=1}^{k-1} x_j) = 0, \quad \forall i \qquad (*)$$

$$\overset{\sum_i}{\rightsquigarrow} -n_k \sum_{i=1}^{k-1} x_i + (\sum_{i=1}^{k-1} n_i)(1 - \sum_{i=1}^{k-1} x_i) = 0$$

$$1 - \sum_{i=1}^{k-1} x_i = \frac{n_k}{n_k + \sum_{i=1}^{k-1} n_i}$$

$$\overset{\text{in } (*)}{\rightsquigarrow} x_i = \frac{n_i(1 - \sum_{j=1}^{k-1} x_j)}{n_k} = \frac{n_i}{n_k + \sum_{i=1}^{k-1} n_i}$$

References

Anonymous.
Partial sorting, May 2013.
Page Version ID: 554578389.

Christopher M. Bishop.
Pattern recognition and machine learning, volume 1.
springer New York, 2006.

Trevor Hastie, Robert Tibshirani, Jerome Friedman, and James Franklin.
The elements of statistical learning: data mining, inference and prediction.
The Mathematical Intelligencer, 27(2):83–85, 2005.

Kevin P. Murphy.
Machine learning: a probabilistic perspective.
The MIT Press, 2012.